

#  
2

IN THE U.S. PATENT AND TRADEMARK OFFICE

JCS25 U.S. PRO  
09/482926  
01/14/00

Applicant(s): BEOM, Jae Young

Application No.:

Group:

Filed: January 14, 2000

Examiner:

For: DEVICE AND METHOD FOR FILTERING ADDED INFORMATION

L E T T E R

Assistant Commissioner for Patents  
Box Patent Application  
Washington, D.C. 20231

January 14, 2000  
0465-0658P-SP

Sir:

Under the provisions of 35 USC 119 and 37 CFR 1.55(a), the applicant hereby claims the right of priority based on the following application(s):

<u>Country</u>	<u>Application No.</u>	<u>Filed</u>
REPUBLIC OF KOREA	859/1999	01/14/99

A certified copy of the above-noted application(s) is(are) attached hereto.

If necessary, the Commissioner is hereby authorized in this, concurrent, and future replies, to charge payment or credit any overpayment to deposit Account No. 02-2448 for any additional fees required under 37 C.F.R. 1.16 or under 37 C.F.R. 1.17; particularly, extension of time fees.

Respectfully submitted,

BIRCH, STEWART, KOLASCH & BIRCH, LLP

By: 

JOSEPH A. KOLASCH

Reg. No. 22,463

P. O. Box 747

Falls Church, Virginia 22040-0747

Attachment  
(703) 205-8000  
/sas

*Beck Alewaut Ital*  
*703-205-8006*  
*Jae Young Kim*  
*465-6581*  
*1071*  
JC525 U.S. PAT.  
09/482999  
01/14/00

# 대한민국 특허청

## KOREAN INDUSTRIAL PROPERTY OFFICE

별첨 사본은 아래 출원의 원본과 동일함을 증명함.

This is to certify that the following application annexed hereto  
is a true copy from the records of the Korean Industrial  
Property Office.

출원번호 : 1999년 특허출원 제859호  
Application Number

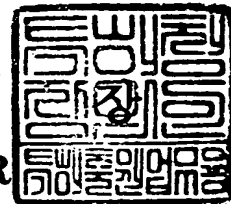
출원년월일 : 1999년 1월 14일  
Date of Application

출원인 : 엘지전자 주식회사  
Applicant(s)

1999년 12월 30일

특허청

COMMISSIONER



【서류명】	출원서
【권리구분】	특허
【수신처】	특허청장
【참조번호】	2
【제출일자】	1999.01.14
【국제특허분류】	H04N
【발명의 명칭】	부가 정보 필터링 방법
【발명의 영문명칭】	Method for PSI filtering
【출원인】	
【명칭】	엘지전자 주식회사
【출원인코드】	1-1998-000275-8
【대리인】	
【성명】	김용인
【대리인코드】	9-1998-000022-1
【포괄위임등록번호】	1999-001100-5
【대리인】	
【성명】	심창섭
【대리인코드】	9-1998-000279-9
【포괄위임등록번호】	1999-001099-2
【발명자】	
【성명의 국문표기】	범재영
【성명의 영문표기】	BEOM, Jae Young
【주민등록번호】	720312-1559217
【우편번호】	137-130
【주소】	서울특별시 서초구 양재동 86-9 강남빌라 402호
【국적】	KR
【취지】	특허법 제42조의 규정에 의하여 위와 같이 출원합니다. [ 김용인 (인) 대리인 심창섭 (인)
【수수료】	
【기본출원료】	20 면 29,000 원
【가산출원료】	0 면 0 원

1019990000859

1999/12/31

【우선권주장료】	0	건	0	원
【심사청구료】	0	항	0	원
【합계】	29,000	원		

**【요약서】****【요약】**

MPEG-2 시스템 레이어에서 트랜스포트 스트림을 구성하는 프로그램 사양 정보 (PSI)를 처리하는 부가정보 필터링 방법에 관한 것으로서, 특히 하나 이상의 섹션들이 조합되어 부가 정보가 규정하는 테이블을 구성하며, 상기 테이블 ID들을 저장하는 메모리와 각 테이블 ID별로 섹션의 버전 번호를 저장하는 메모리가 구비되어, 섹션이 입력되면 상기 섹션에 포함된 테이블 ID와 메모리에 저장된 테이블 ID의 매치 여부를 판별하고, 테이블 ID가 매치된다고 판별되면 상기 입력되는 섹션에 포함된 버전 번호와 상기 해당 테이블 ID의 버전 메모리에 저장된 버전 번호의 일치 여부를 판별한 후 일치한다고 판별되면 현재 입력되는 섹션을 스킵하고, 일치하지 않는다고 판별되면 현재 섹션을 입력받아 처리함으로써, 하드웨어 즉, 메모리의 크기와 복잡도를 줄이고, 필요한 섹션을 선택하는 대신 불필요한 섹션을 버림으로써, 효율적인 시스템을 구성할 수 있다. 또한 필터 설정에 걸리는 시간 때문에 처리하여야 할 섹션을 놓치거나 필터를 낭비하는 것을 방지할 수 있다.

**【대표도】**

도 2

**【색인어】**

정보 필터링

**【명세서】****【발명의 명칭】**

부가 정보 필터링 방법{Method for PSI filtering}

**【도면의 간단한 설명】**

도 1은 종래의 섹션 필터의 예를 보인 도면

도 2는 본 발명에 따른 부가 정보 필터링 방법을 하드웨어로 구현한 도면

도 3은 본 발명의 제 1 실시예에 따른 부가 정보 필터링 방법을 수행하기 위한 흐름도

도 4는 본 발명의 제 2 실시예에 따른 부가 정보 필터링 방법을 수행하기 위한 흐름도

**【발명의 상세한 설명】****【발명의 목적】****【발명이 속하는 기술분야 및 그 분야의 종래기술】**

- <5> 본 발명은 엠펙-2(MPEG-2) 트랜스포트 디코더에 관한 것으로서, 특히 MPEG-2 시스템 레이어에서 트랜스포트 스트림을 구성하는 프로그램 사양 정보(Program specification information ; PSI)를 처리하는 부가정보 필터링 방법에 관한 것이다.
- <6> 동화상 압축 방법으로 널리 알려진 MPEG-2의 경우 동영상 데이터 이외에도 음성 데이터 압축 및 이들을 전송하기에 적합하게 바꾸는 시스템 레이어(system layer)에 대해서도 규정하고 있다.
- <7> 상기 시스템 레이어에는 두 가지의 방식을 규정했는데 하나는 트랜스포트 스트림

(transport stream ; TS)이고, 다른 하나는 프로그램 스트림(program stream ; PS)이다.

- <8> 여기서, PS 방식은 복수의 PES(packetized elementary stream)를 그룹화해서 팩을 구성하는데 반해, TS 방식은 역으로 PES를 분할하여 복수의 트랜스포트 패킷에 실어 전송한다.
- <9> 즉, 상기 TS는 다수의 비디오와 오디오 개별 비트열을 전송하고 있기 때문에 복수의 프로그램 중에서 어느 프로그램을 골라 어느 패킷을 취하여 어떻게 복호해야 하는지에 관한 정보가 필요하게 된다. 이들 프로그램 사양 정보를 총칭해서 PSI라 부른다. 상기 PSI는 지정된 식별 코드를 가진 패킷이나 일차적인 PSI에서 가리키는 패킷등에 의해 전송된다. 상기 MPEG-2 시스템 레이어에서 PSI는 TS를 구성하는 요소 중의 하나이다.
- <10> 이러한 TS의 복호.재생에 있어서는 복수 프로그램 중에서 하나를 선택하고 다음에 그 프로그램의 복호.재생을 위해 필요한 개별 비트열의 트랜스포트 패킷의 패킷 인식 번호(PID)를 알아야 한다. 이 후, 이들 개별 비트열의 파라미터 정보나 연계 정보를 알아야 한다. 이와 같은 다단계 동작을 위하여 다수개의 부가 정보(PSI) 테이블이 필요하며, 이들 테이블은 섹션이라 불리는 데이터 구조에 의해 전송된다.
- <11> 상기 다수개의 테이블 중 PAT(Program Association Table)는 PID=0인 패킷에 의해 전송되는 특수 정보이다. 각 프로그램 번호마다 그 프로그램의 구성 요소를 기술하는데, 테이블(프로그램 맵 테이블)을 전송하는 트랜스포트 패킷의 PID를 가리킨다.
- <12> 상기 프로그램 맵 테이블(Program Map Table)은 프로그램 식별 번호와 프로그램을 구성하는 비디오, 오디오 등의 개별 비트열이 전송되고 있는 트랜스포트 패킷의 PID 리스트와 부속 정보를 기술하고 있다.

- <13> 이때, 비디오나 오디오 데이터를 부호화한 ES(elementary stream)를 다루는 PES(packetized elementary system)와는 달리, PSI의 경우 프로그램에 관련된 정보를 포함하는데, 빠른 액세스가 가능하게 하기 위해 리던더시(redundancy)를 갖게된다. 즉, 언제든지 재빨리 복호화를 시작할 수 있도록 같은 정보를 자주 보내게 된다.
- <14> 일 예로, ATSC(Advanced Television Systems Committee) 규격에서는 PAT의 경우 적어도 0.1초, PMT는 0.4초 이내에 정보를 보내주게 되어 있다. 즉, PAT가 안오면 어느게 비디오인지 오디오인지 알 수 없어 디코딩에 문제가 생길 수 있기 때문이다.
- <15> 그러나, 대부분의 경우에 있어서 이 정보들은 변화가 없기 때문에 이전에 보내주었던 것과 같은 정보를 보내주게 된다. 그러므로, 보통의 트랜스포트 디코더는 이렇게 중복되는 데이터들을 효과적으로 처리하기 위한 알고리즘을 가지고 있으며 널리 알려진 방법으로 디지털 비디오 방송(Digital Video Broadcasting ; DVB) 규격에서 규정하고 있는 섹션 필터링(section filtering)이 있다.
- <16> 즉, MPEG-2에서 PSI는 4종류의 테이블을 규정하고 있는데 이들 모두 섹션이라는 기본 단위를 가지며 하나 이상의 섹션들이 조합되어 하나의 테이블을 구성하게 된다. 이때, 하나의 정보를 하나의 섹션에 넣어서 전송할 수도 있고 복수개의 섹션에 넣어 전송할 수도 있다.
- <17> 상기 섹션은 하드웨어나 소프트웨어의 편의를 위해 비슷한 포맷의 신택스를 갖고 있는데 그 중 하나인 사적 섹션(private section)의 신택스는 하기와 같다.
- <18> private\_section() {
- <19> table\_id

```
<20> section_syntax_indicator  
  
<21> private_indicator  
  
<22> reserved  
  
<23> private_section_length  
  
<24> if(section_syntax_indicator=='0'){  
  
<25> for(i=0; i<N; I++){  
  
<26> private_date_byte  
  
<27> }  
  
<28> }  
  
<29> else {  
  
<30> table_id_extension  
  
<31> reserved  
  
<32> version_number  
  
<33> current_next_indicator  
  
<34> section_number  
  
<35> last_section_number  
  
<36> for(i=0; i<private_section_length-9;  
  
<37> i++){  
  
<38> private_date_byte;  
  
<39> }
```

<40> CRC\_32

<41> }

<42> }

<43> 여기서, 섹션 필터링이란 섹션의 처음 몇 개의 데이터 바이트에 대해 필터를 두어 섹션 내의 특정 위치의 데이터가 설정된 값과 동일한 값을 갖는 섹션만을 처리하는 방법이다.

<44> 즉, 필터링할 위치에 있는 각각의 데이터 바이트에 대해 매치 데이터(match data)와 마스크 바이트(mask byte)(또는 비트)를 설정하여 두고 마스크가 되지 않은 데이터에 대해서만 매치 데이터와 비교하여 같은 경우에만 지정된 위치에 저장한다.

<45> 도 1은 8 바이트 섹션 필터의 한 예로서, 마스크 비트가 1인 경우 해당 바이트(또는 비트)의 섹션 데이터와 매치 데이터를 비교하여 같은 경우의 섹션만 처리하게 된다. 도 1에서 A 타입의 섹션 필터는 마스크가 비트 단위로 설정되어 있는 경우이고, B 타입의 섹션 필터는 마스크가 바이트 단위로 설정되어 있는 경우이다.

<46> 예를 들면, 도 1의 (a)의 섹션 A의 데이터와 (c)의 마스크 비트가 1인 A 타입 섹션 필터의 매치 데이터를 비교했을 때 두 데이터들이 모두 일치함을 알 수 있다. 이 경우에는 매치되었다고 판별하고 섹션 A의 데이터를 받아들여 지정된 위치에 저장한다. 여기서, 섹션 A의 데이터와 (d)의 마스크 바이트가 1인 B 타입 섹터 필터의 매치 데이터와는 일치하지 않으므로, B타입 섹션 필터에서는 섹션 A의 데이터를 받아들이지 않는다. 즉, (a)의 섹션 A와 (c)의 A 타입 섹션 필터가 매치를 이루고 (b)의 섹션 B와 (d)의 B

타입 섹션 필터가 매치를 이루고 있음을 알 수 있다.

<47> 이렇게 하여 전송되어오는 PSI 섹션중에서 원하는 것을 선택하여 처리할 수가 있게 된다. 실제로 DVB 규격은 8바이트 이상의 섹션 필터를 32개 이상 갖추는 것을 규정하고 있다.

【발명이 이루고자 하는 기술적 과제】

<48> 그러나, 이러한 섹션 필터링은 많은 양의 하드웨어를 사용하며, 호스트가 직접 필터를 설정하므로 호스트의 부담이 커지고 또한, 여기에 소요되는 시간 때문에 처리해야 할 섹션을 놓치거나 필터를 낭비하는 경우가 생기는 단점이 있다.

<49> 즉, 도 1에서는 섹션\_길이, 미사용 비트등 불필요한 부분의 데이터에 대해서도 마스크를 씌워 비교함으로써, 하드웨어의 낭비를 초래한다.

<50> 본 발명은 상기와 같은 문제점을 해결하기 위한 것으로서, 본 발명의 목적은 일반적인 섹션 필터대신 비중이 큰 특정 위치에만 필터를 적용하여 하드웨어의 크기를 줄이는 부가정보 필터링 방법을 제공함에 있다.

<51> 본 발명의 다른 목적은 필요한 섹션을 선택하는 대신 불필요한 섹션을 버림으로써, 효율적인 시스템을 구성하는 부가정보 필터링 방법을 제공함에 있다.

【발명의 구성 및 작용】

<52> 상기와 같은 목적을 달성하기 위한 본 발명에 따른 부가정보 필터링 방법은, 하나 이상의 섹션들이 조합되어 부가 정보가 규정하는 테이블을 구성하며, 상기 테이블 ID들을 저장하는 메모리와 각 테이블 ID별로 섹션의 버전 번호를 저장하는 메모리가 구비되어 필요한 섹션만을 받아들이는 부가정보 필터링 방법에 있어서, 섹션이 입력되면 상기

섹션에 포함된 테이블 ID와 메모리에 저장된 테이블 ID의 매치 여부를 판별하는 단계와, 상기 단계에서 테이블 ID가 매치한다고 판별되면 상기 입력되는 섹션에 포함된 버전 번호와 상기 해당 테이블 ID의 버전 메모리에 저장된 버전 번호의 일치 여부를 판별하는 단계와,

<53> 상기 단계에서 테이블 ID가 매치되지 않거나 상기 단계에서 두 버전 번호가 일치한다고 판별되면 현재 입력되는 섹션을 스킵하는 단계와, 상기 단계에서 두 버전 번호가 일치하지 않는다고 판별되면 현재 섹션을 입력받아 처리하는 단계를 포함하여 이루어지는 것을 특징으로 한다.

<54> 상기 섹션 스킵 단계는 현재 입력되는 섹션의 버전 번호와 메모리에 저장된 버전 번호가 동일하고 상기 버전 번호가 마스크 인에이블이면 현재 입력되는 섹션을 스킵하는 것을 특징으로 한다.

<55> 상기 테이블 ID 비교 단계에서 현재 입력되는 섹션의 테이블 ID와 메모리에 저장된 테이블 ID가 매치된다고 판별되면 현재 처리하고 있는 버전의 테이블이 완성되는지를 판별하는 단계와, 상기 단계에서 테이블이 완성되었다고 판별되면 상기 섹션의 버전 번호에 대해 마스크 인에이블시키는 단계를 더 포함하여 이루어지는 것을 특징으로 한다.

<56> 본 발명의 다른 목적, 특징 및 잇점들은 첨부한 도면을 참조한 실시예들의 상세한 설명을 통해 명백해질 것이다.

<57> 이하, 본 발명의 바람직한 실시예를 첨부도면을 참조하여 상세히 설명한다.

<58> 본 발명은 테이블 ID와 버전 번호만을 필터에 적용하여 하드웨어 특히, 메모리의 크기와 복잡도를 줄인다. 여기서, 테이블 ID는 테이블마다 할당되어 있으며 섹션마다 들

어있다.

- <59>      본 발명은 테이블 ID가 같고 버전 번호가 다른 데이터에 대해서만 받아들인다.
- <60>      도 2는 이러한 본 발명에 따른 부가정보 필터링 방법을 하드웨어로 구현한 예이고,  
           도 3은 본 발명의 제 1 실시예에 따른 부가정보 필터링 방법의 개략적인 흐름도이다.
- <61>      도 2를 보면, 섹션의 테이블 ID를 저장하는 테이블 ID 메모리(22)와 각 테이블 ID  
           별로 마스킹 인에이블 비트(E)와 마스크할 버전을 위한 메모리(21)가 있다.
- <62>      만일, 섹션(20)이 입력되면 섹션의 시작을 찾아야 하는데(단계 301), PSI 페이로드  
           를 갖는 TS 패킷에서 페이로드 유닛\_스타트\_인디케이터가 '1'이면 포인터 필드를 보고  
           섹션의 처음을 알 수 있다. 또한 하나의 섹션이 끝난 후 다음 데이터가 0xFF가 아니면  
           또다른 섹션이 시작된다.
- <63>      이때, 섹션(20)의 첫 바이트는 테이블 ID이다. 그러므로, 상기 테이블 ID와 테이블  
           ID 메모리(22)에 저장되어 있는 테이블 ID들과 비교하여 매치되는지를 확인한다(단계  
           302).
- <64>      만일, 매치되면 매치된 테이블 ID에 할당된 버전 번호를 상기 테이블 ID의 버전 메  
           모리(21)로부터 읽어오고 일치하지 않으면 현재 섹션을 스킵하여 버린다.
- <65>      그리고 나서, 섹션에 포함된 버전 번호와 비교하는데 먼저, 마스킹 인에이블 신호  
           (E)가 1인지를 판별하여 1이면 비교한다(단계303).
- <66>      이때, 현재 섹션의 버전이 버전 메모리(21)의 버전과 동일하고 마스크 인에이블이  
           면 현재 버전은 마스킹되어 있다. 즉, 이미 동일한 데이터가 입력되어 저장되어 있다.  
           따라서, 이때는 현재 섹션을 스킵하여 버린다(단계 304). 여기서, 마스킹 인에이블 신호

(E)는 입력되는 섹션을 저장할 것인지를 결정하기 위한 플래그이다.

<67>       상기 단계 303에서 현재 섹션의 버전이 버전 메모리(21)의 버전과 다르고 마스크 인에이블이면 현재 섹션을 저장하거나 처리한다(단계 305).

<68>       도 4는 본 발명의 부가 정보 필터링 방법의 다른 실시예를 나타낸 흐름도로서, 입력되는 섹션의 시작을 찾는 방법은 도 3과 동일하다(단계 401).

<69>       그러므로, 상기 단계 401에서 섹션의 시작을 찾으면 섹션에 포함되는 테이블 ID와 메모리(22)에 저장된 테이블 ID가 매치되는지를 판별한다(단계 402).

<70>       상기 단계 402에서 두 테이블 ID가 매치되고 마스크 인에이블 비트(E)가 1이면 해당 테이블 ID의 버전 메모리(21)에 저장된 버전 번호와 상기 섹션의 버전 번호를 비교한다(단계 403).

<71>       상기 단계 403에서 현재 섹션의 버전이 버전 메모리(21)의 버전과 동일하고 마스크 인에이블 비트(E)가 1이면 현재 버전은 마스킹되어 있으므로 현재 섹션을 스킵하여 버린다(단계 403).

<72>       상기 단계 404에서 현재 섹션의 버전이 버전 메모리(21)의 버전과 다르고 마스크 인에이블이면 현재 섹션을 저장하거나 처리한다(단계 405).

<73>       한편, 상기 단계 402에서 테이블 ID가 매치되었다고 판별되면 하나의 테이블이 완성되었는지를 판별한다(단계 406).

<74>       현재 처리하고 있는 버전의 테이블이 모두 입력되어 처리되었거나 더 이상의 섹션이 불필요한 경우에는 테이블이 완성되었다고 본다. 즉, 입력되는 섹션에서 section\_number와 last\_section\_number를 받아서 테이블의 완성을 확인한다.

- <75> 만일, 상기 단계 406에서 테이블이 완성되었다고 판별되면 상기 테이블의 테이블 완성 비트(C)를 1로 셋트하고, 버전 번호 자동 셋팅 비트(A)가 1이면 입력되는 버전 번호를 버전 메모리(21)에 저장한다.
- <76> 여기서, 상기 테이블 완성 비트(C)는 임의의 버전을 갖는 섹션을 모두 처리한 후에 입력되는 같은 버전의 섹션을 처리하지 않도록 자동적으로 설정하는 플래그이다. 또한, 버전 번호 자동 셋팅 비트(A)는 새로운 버전을 가진 섹션이 입력되었을 경우 버전 메모리에 저장된 버전을 새로운 버전으로 자동으로 업데이트할 수 있게 하는 플래그이다.
- <77> 즉, 입력되는 버전 번호를 메모리(21)에 저장할 때 테이블 완성 비트(C)가 1이면 마스크 인에이블 비트(E)를 1로 셋팅시킨 후 버전 번호를 저장한다.
- <78> 본 발명에서는 테이블이 완성되었는지를 체크하는 알고리즘은 제안하지 않았다.
- <79> 그러나, 테이블이 섹션 하나로 이루어진 경우 즉, last\_section\_number가 0x00인 경우에는 현재 입력된 섹션이 처리되면 테이블이 완성되므로 쉽게 구현된다.
- <80> 또한, 본 발명에서는 새로운 버전이 입력되면 버전 메모리를 새로운 버전 값으로 업데이트할 수 있게 하는 플래그를 두어 빠른 시간에 버전이 급변하는 경우에 호스트가 관여하지 않아도 대응할 수 있게 하였다.
- <81> 이상에서 살펴본 바와 같이, 본 발명은 종래의 섹션 필터링에 비해 하드웨어가 크게 감소한다.
- <82> 예를 들어, 마스크 바이트를 갖는 종래의 8 바이트의 섹션 필터와 본 발명의 메모리 크기를 비교해보자.

- <83> 먼저, N개의 섹션 필터를 갖는다고 한다면 종래의 메모리 사이즈  $M_{sec}$ 은
- <84>  $M_{sec} = N \times 8 \text{ 바이트} \times 2 = 128N$ 이 되고,
- <85> 본 발명에서 N개의 테이블 ID에 대해 버전 마스킹을 한다면 메모리 사이즈  $M_{ver}$ 은
- <86>  $M_{ver} = N \times 8 \text{비트} \times 2 = 16N$ 이 된다.
- <87> 즉, 메모리 사이즈가 1/8로 줄어든다는 것을 알 수 있다.
- <88> 또한, 본 발명은 마스크할 버전을 하드웨어에서 자동으로 세팅할 수 있으므로 연속적으로 버전이 업데이트된 섹션이 입력되는 경우에도 호스트의 성능과 무관하게 섹션을 놓치지 않고 처리할 수 있다.
- <89> 예를 들어, 동일한 임의의 테이블 ID를 갖고 버전 번호가 0x00, 0x01, 0x02인 3개의 섹션이 있고, 각 섹션이 하나의 테이블을 구성한다고 하자. 먼저 버전 0x00인 섹션이 지속적으로 들어오다가 버전 0x01인 섹션이 하나 들어오고 다음에 이어서 버전 0x02인 섹션이 계속 들어왔을 때 이 세 버전을 처리하는 과정을 살펴보기로 하자.
- <90> 종래의 섹션 필터링의 경우, 버전 0x00인 섹션은 일단 처리하고 버전 0x01를 처리할 수 있도록 필터를 설정해 놓는다. 그리고, 버전 0x01인 섹션이 들어오면 저장하고 호스트가 처리할 수 있도록 한다. 호스트는 저장된 섹션을 읽어보고 디코딩한 후 버전 0x02를 처리할 수 있도록 필터를 설정한다. 그러나 위의 경우에 버전 0x02를 처리하도록 필터를 설정하는데 걸리는 시간이 길어져서 버전 0x02인 첫 섹션을 놓칠 수가 있다.
- <91> 그러나, 본 발명에서는 버전 0x01이 들어오는 즉시 하드웨어에서 버전 0x01인 섹션을 마스킹하기 때문에 버전이 0x02인 다음 섹션을 놓치지 않고 처리할 수가 있게 된다.

<92> 두 방법이 이런 차이점을 나타내는 것은 종래의 섹션 필터링의 경우 필요한 섹션을 선택할 수 있도록 설계되고, 본 발명은 버전 마스킹을 불필요한 섹션을 버리도록 설계한 데 기인한다.

<93> 또한, 본 발명은 32개의 섹션 필터를 사용하지 않고도 MPEG PSI를 효과적으로 처리할 수 있으므로 MPEG TS를 시스템 레이어로 사용하는 트랜스포트 디코더에서 사용될 수 있다.

<94> 또한, 본 발명은 다른 시스템 레이어의 시스템 정보 분야에 적용하여 하드웨어를 줄이거나 호스트의 개입없이 지능적인 디코딩을 하는 효과를 볼 수 있다.

#### 【발명의 효과】

<95> 이상에서와 같이 본 발명에 따른 부가 정보 필터링 방법에 의하면, 일반적인 필터 대신 비중이 큰 특정 위치(테이블 ID, 버전 번호)에만 필터를 적용하여 하드웨어 즉, 메모리의 크기와 복잡도를 줄이고, 필요한 섹션을 선택하는 대신 불필요한 섹션을 버림으로써, 효율적인 시스템을 구성할 수 있다. 또한 필터 설정에 걸리는 시간 때문에 처리하여야 할 섹션을 놓치거나 필터를 낭비하는 것을 방지할 수 있다. 또한, 트랜스포트 디코더에서 입력되는 섹션의 입력 여부를 판단하여 불필요한 섹션은 버리고 필요한 섹션만 받아들이므로 그만큼 호스트에 로드가 적게 걸리게 되어 호스트의 성능을 향상시킬 수 있다.

**【특허청구범위】****【청구항 1】**

하나 이상의 섹션들이 조합되어 부가 정보가 규정하는 테이블을 구성하며, 상기 테이블 ID들을 저장하는 메모리와 각 테이블 ID별로 섹션의 버전 번호를 저장하는 메모리가 구비되어 필요한 섹션만을 받아들이는 부가정보 필터링 방법에 있어서,

섹션이 입력되면 상기 섹션에 포함된 테이블 ID와 메모리에 저장된 테이블 ID의 매치 여부를 판별하는 단계와,

상기 단계에서 테이블 ID가 매치한다고 판별되면 상기 입력되는 섹션에 포함된 버전 번호와 상기 해당 테이블 ID의 버전 메모리에 저장된 버전 번호의 일치 여부를 판별하는 단계와,

상기 단계에서 테이블 ID가 매치되지 않거나 상기 단계에서 두 버전 번호가 일치한다고 판별되면 현재 입력되는 섹션을 스킵하는 단계와,

상기 단계에서 두 버전 번호가 일치하지 않는다고 판별되면 현재 섹션을 입력받아 처리하는 단계를 포함하여 이루어지는 것을 특징으로 하는 부가 정보 필터링 방법.

**【청구항 2】**

제 1 항에 있어서, 상기 섹션 스킵 단계는

현재 입력되는 섹션의 버전 번호와 메모리에 저장된 버전 번호가 동일하고 상기 버전 번호가 마스크 인에이블이면 현재 입력되는 섹션을 스킵하는 것을 특징으로 하는 부가정보 필터링 방법.

**【청구항 3】**

제 1 항에 있어서,

상기 테이블 ID 비교 단계에서 현재 입력되는 섹션의 테이블 ID와 메모리에 저장된 테이블 ID가 매치된다고 판별되면 현재 처리하고 있는 버전의 테이블이 완성되는지를 판별하는 단계와,

상기 단계에서 테이블이 완성되었다고 판별되면 상기 섹션의 버전 번호에 대해 마스크 인에이블시키는 단계를 더 포함하여 이루어지는 것을 특징으로 하는 부가정보 필터링 방법.

**【청구항 4】**

제 1 항에 있어서, 상기 테이블 ID 비교 단계는

트랜스포트 패킷에서 페이로드\_신택스\_인디케이터가 '1'이면 포인터 필드를 보고 입력되는 섹션의 시작을 판단하는 것을 특징으로 하는 부가정보 필터링 방법.

**【청구항 5】**

제 1 항에 있어서, 상기 테이블 ID 비교 단계는

하나의 섹션이 끝난 후 다음 데이터가 0xFF가 아니면 또다른 섹션이 시작된다고 판별하는 것을 특징으로 하는 부가 정보 필터링 방법.

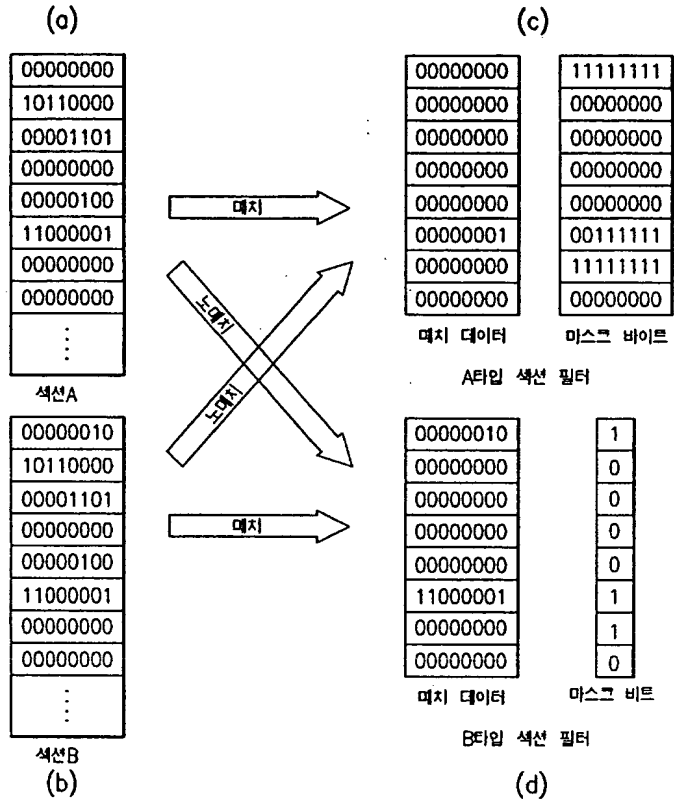
**【청구항 6】**

제 1 항에 있어서, 상기 두 버전 번호가 일치하지 않는다고 판별되면 현재 섹션을 입력받아 처리하는 단계는

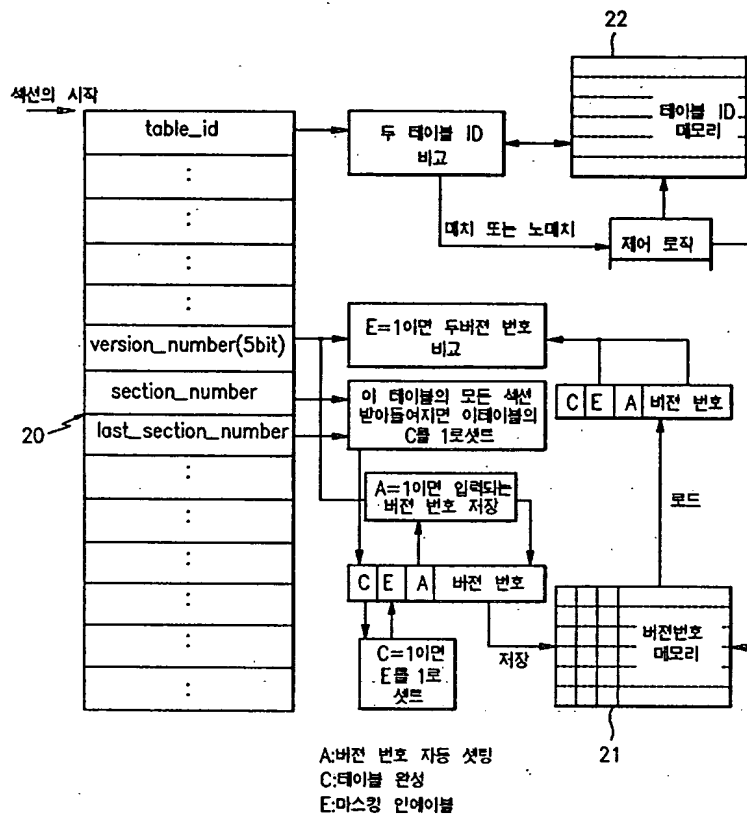
새로운 버전 번호가 입력되었을 때 버전 메모리에 저장된 버전값을 자동적으로 셋팅해주는 것을 특징으로 하는 부가 정보 필터링 방법.

## 【도면】

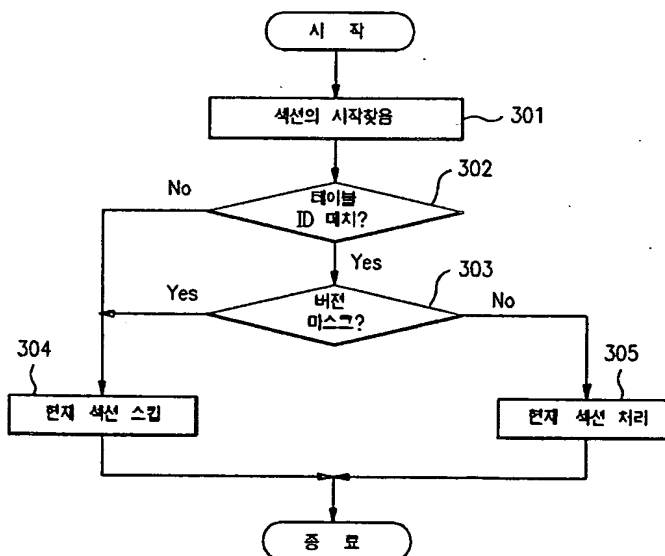
【도 1】



【도 2】



【도 3】



【도 4】

